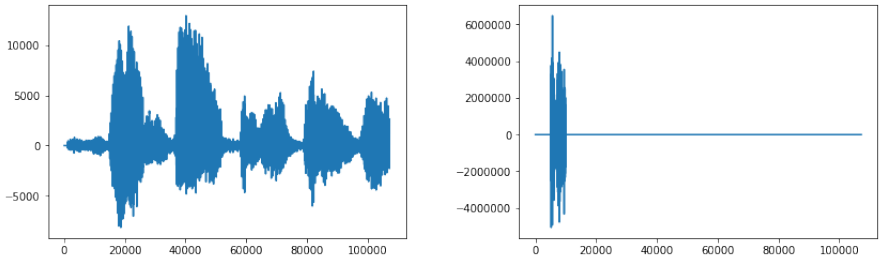
Assignment 3: Sounds Good

2020 Fall EECS205002 Linear Algebra

Due: 2020/12/16

In physics, sound is a vibration that propagates as an acoustic wave, through a transmission medium such as a gas, liquid or solid. Because of their wave property, most people analyze and manipulate the sound signals using Fourier Transform (FT). In this assignment, we study a similar but simpler method, called Discrete Cosine Transform (DCT), which also transforms sound signals from the time domain to the frequency domain. Figure 1 shows an example of the wave form for my recorded sound and the corresponding signal in the frequency domain. There are two interesting properties observed from the figures. First, the number of data (x-axis) in both domains are the same. Second, most data in the frequency domain are 0.



(a) Time domain (b) Frequency domain

Figure 1: Sound signals in the time domain and the frequency domain.

The idea of DCT is *interpolation*, which we had learned in assignment 2. Different from that in assignment 2, where the basis are polynomials 1, x, and x^2, the basis of DCT are the cosine functions $1/\sqrt{2}, \cos(x), \cos(2x), \ldots$. Following the same principle, we want to find the coefficients $a_0, a_1, a_2 \ldots$ so that the linear combination

$$F(x) = a_0/\sqrt{2} + a_1 \cos(x) + a_2 \cos(2x) + \cdots$$

can pass all the given points $(x_0, y_0), (x_0, y_0), \ldots, (x_{n-1}, y_{n-1})$, which is equivalent to solve

$$F(x_i) = y_i \text{ for } i = 0, 1, \ldots, n-1. \tag{1}$$

Since there are n equations, we only need n variables $a_0, a_1, \ldots, a_{n-1}$ to make every equality hold. So the required basis are $1/\sqrt{2}, \cos(x), \ldots, \cos((n-1)x)$.

As we have learned in assignment 2, equation (1) can be expressed as a linear system

$$Xa = y, \tag{2}$$

where

$$X = \begin{bmatrix} 1/\sqrt{2} & \cos(x_0) & \cdots & \cos((n-1)x_0) \\ 1/\sqrt{2} & \cos(x_1) & \cdots & \cos((n-1)x_1) \\ \vdots & \vdots & \ddots & \vdots \\ 1/\sqrt{2} & \cos(x_{n-1}) & \cdots & \cos((n-1)x_{n-1}) \end{bmatrix} \tag{3}$$

$$a = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{bmatrix} \text{ and } y = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \end{bmatrix}.$$

Last, if x_i are taken at special points,

$$x_i = \frac{\pi}{n}\left(i + \frac{1}{2}\right) \text{ for } i = 0, 1, \ldots, n-1. \tag{4}$$

the matrix X will have some special properties, which you will see in the assignment.

1 Assignment

1. (20%) Let $n = 4$, prove $X^T X = 2I$ for the X defined in (3) and the x_i taken at the points specified in (4). Therefore, the inverse of X is $\frac{1}{2}X^T$, and the solution in (2) is $a = \frac{1}{2}X^T y$. (You may need to reference the trigonometric identities online, such as `https://en.wikipedia.org/wiki/List_of_trigonometric_identities`.)

2. (20%) For $n = 4$, consider the least square problem

$$\min_{a_1, a_2} \sum_{i=0}^{3} \left(a_1 \cos(x_i) + a_2 \cos(2x_i) - y_i\right)^2 \tag{5}$$

where x_i are taken at the points specified in (4). Show the solution is

$$\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \frac{1}{2} X[:, 2:3]^T y \tag{6}$$

which are the same as the solution of a_1 and a_2 in question 1. Why it is different from the solution of normal equation? (You may reference Theorem 5.5.6.)

3. (20%) Record a 2 second sound wave, and perform `scipy.fftpack.dct` on it. Filter out part of the transformed signals, and transform the data in the frequency domain to time domain using `scipy.fftpack.idct`. Store the filtered waves. Plot the waves before and after the processing, and compare their differences.

4. (20%) Frequency is the number of occurrences of a repeating event per unit of time ($f = 1/T$). Different cosine functions are corresponding to different frequencies, as shown in Figure 2. Using DCT, you can remove some of the frequencies by setting their coefficients zero (this is called filtering). Discuss the effects on the sound signals after you remove the low frequency part, and the effects on the sound signals after you remove the high frequency part.

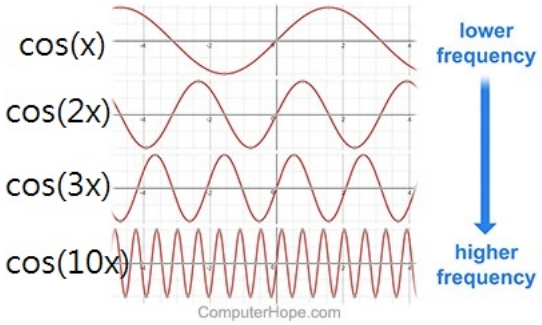


Figure 2: Frequency and cosine function.

5. (20%) This is a competition problem again. The second property of DCT that most data in the frequency domain are zeros can be used to compress the data. Design and implement a compression algorithm based on only DCT/IDCT to compress sounds. For instance, you may store only the nonzero parts in the DCT and their indices. It is fine if your compression algorithm is lossy (date cannot be fully recovered). The score of this problem will be judged by

$$\text{score} = \frac{\text{compressed data size (byte)} \times (1 + \text{error})}{\text{original data size (byte)}}.$$

It is the smaller the better. The compressed data size can be measured by

$$\text{number of stored data} \times \text{number of bytes used for each datum.}$$

The error is defined as

$$\text{error} = \frac{1}{N} \sum_{i=0}^{N} |s_i - t_i|,$$

where $s_0, s_1, \ldots, s_{N-1}$ is the original sound signal, and $t_0, t_1, \ldots, t_{N-1}$ is the recovered signal.

2 Submission

1. Write a report in a PDF file that includes (1), (2), (3), (4), and (5). For question (3), attach the sound file, plots, and your discussion. For question (4), give your settings (how to get the lower part/higher part) and your discussion. For question (5), explain your algorithm and implementation, and the analysis on compressed data size and error.

2. Python code of your implementation of (3), (4), (5).

3. Zip them and submit to iLMS system